05

Setter

Transcripción

[00:00] Ustedes deben estarse preguntando quizás por qué es que en el método obtener saldo yo me he estado confundiendo un poco poniéndole get y después lo borraba. Es muy fácil. Es una costumbre basada en una buena práctica y en una convención.

[00:20] En el caso de Java, ya estamos aquí encapsulando saldo. Nosotros ya que vamos a definir un método para obtener el saldo, para obtener ese valor a través de un método, por convención esos métodos son los llamados métodos get. ¿Qué quiere decir? Él va a hacer lo mismo pero prácticamente el verbo va a ser en inglés, él va a ser un get saldo. ¿Qué significa esto?

[00:51] Por convención Java te dice: "Okay, si tú vas a definir un método para obtener el valor de una variable privada, entonces ese método debe comenzar con get". Es por convención. ¿Qué es una convención? Que simplemente una forma de llamar las cosas o de hacer las cosas que ya está padronizada por toda la comunidad y por todos los programadores Java.

[01:36] Por lo cual, cualquier programador que vea aquí get saldo, este es un método get, es un Getter y yo ya sé exactamente lo que él está haciendo y lo que él debería hacer. Y aquí vemos que obtener saldo ya me está saliendo subrayado con rojo, me está detectando un error. ¿Por qué? Porque yo he cambiado. Voy a cerrar aquí, porque yo he cambiado el nombre del método.

[01:53] ¿Cómo podría hacer entonces para actualizar ese nombre del método? Eclipse nos da una utilidad muy interesante, que aquí le damos, seleccionam

el método, doble clic y le damos a refactor. Y en refactor existe la primera opción que es rename. Le damos a rename y en rename nosotros podemos modificar el nombre de este método.

[02:22] ¿Entonces qué vamos a hacer? Vamos a cambiarlo por el get, le damos enter y él nos va a decir: "Oye, por si acaso, aquí también tienes el mismo método con el mismo nombre que tú quieres cambiar. ¿Deseas que aquí lo cambiemos?" Yo le voy a decir que sí. Continuamos y si vamos a prueba acceso, ya cambió automáticamente a getSaldo.

[02:45] Y ahora de igual forma, ya hemos visto el caso de saldo, la variable saldo. ¿Será que necesitamos hacer lo mismo con agencia y número? Es decir, yo puedo tener una agencia negativa, ¿será que puedo tener un número de cuenta negativo? Lo más seguro es que por la red del negocio la respuesta sea no. Entonces, de igual manera vamos a llamar aquí a nuestra palabra reservada private para decirle: "estos campos también son privados y nadie va a poder acceder directamente a ellos".

[03:22] ¿Pero qué sucede ahora? Yo quiero asignar un valor agencia y yo sí quiero asignarle un valor a número. Con saldo yo no necesito intervenir directamente con la variable porque para eso tengo misión métodos depositar y retirar, pero para el caso de agencia y número yo sí necesito intervenir con ellos directamente. ¿Qué se hace en ese caso? Ahí llega el método set.

[03:48] Si get es para obtener, set es para asignar, y podría ser por ejemplo setAgencia y si yo le voy a asignar valor a esa variable, ¿qué debería hacer? Mandarle un argumento a ese método con el valor que yo le quiero poner. En ese caso es un tipo entero y va a ser nuevaAgencia. Abrimos llaves, el método tiene que ser público y en este caso yo no deseo retornar alguna cosa cuando asigno esa nuevaAgencia.

[04:32] Yo podría retornar un boolean quizás, pero creo que este no es el caso, creo que solamente voy a asignar el valor y eso es todo, entonces le voy a dar

un void. Acuérdense que void es un tipo de método que no retorna un valor. ¿Qué hago aquí? Algo tan simple como decirle que esta variable agencia va a ser igual a nuevaAgencia.

[05:03] Incluso puedo decirle que aquí el nombre de la variable es simplemente agencia y eso no debería dar ningún error de compilación. ¿Por qué? Como ya vimos cuando explicamos el método this, cuando yo pongo this yo hago referencia a la variable de esta clase. Esta variable de aquí va del lado del argumento

[05:25] . Por lo cual, digamos, voy a cerrar esta clase con error, si yo voy a prueba acceso y necesito decirle que el número de agencia va a ser el número 2 por ejemplo, le voy a decir cuenta.agencia, y vemos que yo ya no tengo más acceso al campo agencia, pero tengo acceso a set agencia, si se dan cuenta. Entonces, yo podría darle enter tranquilamente y me va a decir: "setAgencia, por favor dime el número de agencia". Digamos 22.

[06:04] De esta forma yo ya le di un valor a mi variable agencia a través de su modificador de acceso. ¿Y esto en qué me sirve? ¿De qué me sirve crear un método para asignarle valor si yo igual tengo acceso a esta variable? Yo igual puedo ponerle lo que quiera, aquí puedo ponerle un -22 y el código sigue compilando, y de hecho él va a asignarle ese valor negativo a la agencia. Ahí entra la magia del set.

[06:33] ¿Por qué? Porque con setAgencia yo aquí le puedo poner una validación. If. Y le doy. Si agencia es menor que cero, entonces ahí sí yo ahí le voy a decir: "Entonces asígnale ese valor a agencia. Y si no, no hagas nada". Por lo tanto le damos a guardar, guardamos aquí también y le vamos a copiar esta línea y vamos a imprimir agencia.

[07:22] GetAgencia. Nos hemos dado cuenta que no tenemos getAgencia y ya no tenemos acceso a la variable agencia directamente. Hay que escribir el método. ¿Cómo escribimos el método? Muy fácil. Nosotros podemos comenzar

diciéndole getAgen "Ctrl + espacio" y Eclipse ya nos va a dar una sugerencia, nos va a preguntar: "Oye, ¿estás escribiendo un Getter? ¿No querrás un getAgencia?" Le digo que sí y me autogenera el código para yo retornar mi agencia actual.

[08:01] Perfecto. Eso es un atajo de Eclipse. El IDE nos ayuda, nos sugiere algunas plantillas de código que ya tiene para ahorrarnos el trabajo de escribir este código repetitivo. Por lo tanto aquí yo le puedo decir getAgencia y ya me autocompleta automáticamente esto. Le voy a dar guardar, ejecutar, le doy que proceda y agencia es cero, porque es su valor por defecto y él nunca le asignó este valor negativo.

[08:37] ¿Por qué? Porque aquí ya estamos haciendo una validación que si la agencia es mayor que cero, solo en este caso, solo aquí le vamos a setear agencia. Incluso podríamos decirle que else, si no que nos imprima por ejemplo: "No están permitidos valores negativos", por ejemplo. Ejecutamos nuevamente aquí y nos va a decir: "No se permiten valores negativos". Esa es la utilidad real de tener un método set.

[09:19] Podemos definir nuestras reglas del negocio, podemos definir validaciones para una variable en específico y la protegemos de manipulación directamente por parte pues de quien está usando esta variable.